UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/777,361	02/12/2004	Mark Spotswood	ORACL-01312US1	5070

23910 7590 06/05/2009

FLIESLER MEYER LLP
650 CALIFORNIA STREET
14TH FLOOR
SAN FRANCISCO, CA 94108

EXAMINER	
WEI, ZHENG	
ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
06/05/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/777,361	SPOTSWOOD, MARK	
	Examiner	Art Unit	
	ZHENG WEI	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *16 March 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,6,7,9,11-14,16,17,19,31-34 and 43-46* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4, 6, 7, 9, 11-14, 16, 17, 19, 31-34, 43-46* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Remarks

1. This office action is in response to the amendment filed on 03/16/2009.

2. Claims 5, 15 and 35-42 have been canceled.

3. Claims 43-46 have been added.

4. Claims 1 and 11 have been amended.

5. Claims 1-4, 6, 7, 9, 11-14, 16, 17, 19, 31-34 and 43-46 remain pending and have been examined.

Response to Arguments

6. Applicant's arguments filed on 03/16/2009, in particular on pages 11-13, have been fully considered but they are not persuasive. For example:

   - At the page 11, last paragraph, Applicant submits that, in Taylor, the system disclosed therein is used to support programs that are intended only to be executed in remote address space of the remote process. However, as Taylor disclosed at paragraph [0012], "Alternatively, the remote process, remote address space, and client address space may be implemented in a same computer system...[emphasis added]". Therefore, the example of a class loader hierarchy defined by the manifest file/control file/deployment descriptor file and deploying/loading class/module/application using the manifest file in the distributed computer system, can also be implemented in the same computer system. Applicant also submits that Taylor appears to require the

two processes (local and remote) to communicate via shared memory, even when running on the same machine. However, it should be noted that claim language merely recites a server with executing a Java virtual machine that include a system classloader, but does not require any particular processes related to the server and classloaders. Therefore, Taylor still teaches the limitation.

- At page 12, second paragraph, Applicant submits that neither Taylor nor Susarla, when considered alone or in combination, appear to disclose or render obvious hosting multiple isolated software applications within a Java virtual machine, as defined by claim 1, as currently amended.
  However, Peterson (in the record) discloses a concept of an "isolation mode" of WebSphere 4.0 class loading architecture, e.g. "module mode" (one class loader per J2EE module) (see for example, p.5, Fig.3) which includes system classloader, children of the system classloader (WebSphere Applicant Extension class loader) and module level classloaders (one class loader is created for each module in an .ear...). Peterson also discloses another class loading architecture (see for example, p.6, Fig.4) that creates one web application class loader for each web application archive in the J2EE application to allow web application independence (see for example, p.6). Therefore, Taylor, Susarla and Peterson teach the limitation as amended.
- At page 12, last paragraph, the applicant submits that, "in Taylor, the class loader hierarchy therein appears to use a manifest file at the application level.

Similarly, Figures 7 and 8 appear to show that the class loader hierarchy stops at the population instance for the application, but does not appear to proceed to the module level within each population instance. However, as Taylor disclosed at Fig.6 and paragraph [0050], the class loader hierarchy 350 further divides class loader from application/system level to root, factory and component levels (see for example, Fig.6 and paragraph [0050], "The class loader hierarchy 350 divides class loaders into factory, root, and component class loaders. Component class loaders are used to load the variable components of the system, such as services, facility implementations, plug-ins, and dynamically loaded modules").

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claims 1-4, 6, 7, 9, 11-14, 16, 17, 19, 31-34 and 43-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Taylor (Taylor et al., US 2004/0019897 A1) in view of Susarla (Susarla et al., US 6,915,511 B2), and in view of Peterson (Brett Peterson, Understanding J2EE Application Server Class Loading Architectures)

Claim 11:

Taylor discloses a method for loading software applications on a server executing a Java virtual machine that includes a system classloader (root class loaders) (see for example, paragraph [0027], "A client 2 and server 4 would implement a Java Virtual Machine (JVM) to execute Java programs in a Java Runtime Environment (JRE)"), comprising the steps of:

- Providing a server executing a Java virtual machine for storing and running a plurality of software applications (see for example, Fig.5, item 304 Server, item 312 storage, Fig.10, and related text; also see paragraph [0012], "same computer system"; also see paragraph [0027], "A client 2 and server 4 would implement a Java Virtual Machine (JVM) to execute Java programs in a Java Runtime Environment (JRE)")
- providing a plurality of software applications, wherein each of said plurality software applications includes a plurality of deployable modules and classes associated therewith, and wherein the software applications can be customized by a software developer and then deployed to run on the same server (see for example, p.4, paragraph [0047], paragraph [0012} "same computer system");
- providing a control file associated with said software application (see for example, Fig.5 item 325 "population manifest" and related text), wherein said control file can be edited by a software developer (administrator) and specifies a hierarchy of classloaders as children of the system classloader, to

be used with the modules in said software application, and wherein the hierarchy includes a plurality of nested branches that are specified by the software developer to provide namespace separation between two or more of the plurality of software applications or different modules separation between different modules in any one of the software applications (see for example, p.4, paragraph [0048], "An administrator would place… and provide the population manifest 326 indicating the components that will be loaded"; paragraph [0050], "The population manifest 326 is parsed by the container 306 to construct class loader…"; paragraph [0051], "the population manifest 326 may be implemented as multiple population manifests, where each manifest comprises an XML file"; also see Fig.6, Fig.7 and related text);

- upon receiving a request to deploy and run a software application on the server (see for example, Fig.4, step 100,"Receive remote objects form server in response to RMI call on proxy object" and related text)
  - parsing the control file and determining which classloaders are specified therein for the software application being deployed (see for example, paragraph [0050], "The population manifest 326 is parsed by the container 306 to construct class loader…")
  - retrieving a selection of said classloaders according to the hierarchy specified by said control file (see for example, p.4, paragraph [0051], "construct class loaders as necessary to load the components indicated in the manifests"; also see paragraph [0051]); and,

- loading said modules and classes into the Java virtual machine at the server as part of said application according to said hierarchy, (see for example, p.4, paragraph [0051]-[0052], "load the components indicated in the manifests").

Taylor also discloses the module to be reloaded without reloading other modules (see for example, p.4, paragraph [0049]-[0050], "dynamically loaded modules") and but does not explicitly disclose the loading including, if a modules in said software application is being redeployed then loading only the classloaders in the branches for that module, independently of other branches in the hierarchy. However, Susarla in the same analogous art of dynamic redeploying environment discloses a feature about redeploying/reloading only the application class loaders that are specified in the branches for the particular software application/module without loading any of the other branches in the hierarchy (see for example, Fig.4, Fig.5, items 202, 204B, 208, Fig.7, step 306, 308 310 and related text; also see col.11. lines 7-14, "This layer may load all the classes that are visible only to a module but not cross-module"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to redeploy a specific module without reload other modules in the application. One would have been motivated to do so to reduce the number of steps that must be performed as suggested by Susarla (see for example, col.8, lines 1-4, "Using the dynamic class reloading mechanism, only a changed class

and its dependent classes are reloaded, thus limiting the number of files that are affected on the application server").

However, neither Taylor nor Susarla explicitly discloses the hierarchy including providing each of said plurality of software applications with its own application classloader hierarchy so that software application are not aware of classloaders or classes that are assigned to another software application. Peterson, in the same analogous art of J2EE Application server class loading, discloses a hierarchy class loading architecture with isolation mode wherein one class loader is created for each module in an .ear and a module is defined as a web application, an EJB .jar, or a .jar referenced from the Manifest Class-Path of a .war or .jar (same a configuration file) (see for example, Fig.3 and related text). Peterson also discloses a HP-AS 8 MP3 class loading architecture (see for example, Fig.4 and related text), wherein on web application class loader is created as a child of the application class loader (see for example, p.6). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to use Peterson's class loading hierarchy architecture to load the classes of Taylor to JVM in local server to enable application independence (see for example, p.1, first paragraph, "A class loading hierarchy is typically used to enable features such as hot redeployment and application independence.") which is also highly recommended by WebSphere 4.0 documentation (see for example, p.5, lines 3-4, "The WebSphere 4.0 documentation highly recommends using the Module isolation mode in

conjunction with Manifest Class-Path entries")

Claim 12:

Taylor further discloses the method of claim 11 wherein said control file can be modified by a software developer to specify a particular hierarchy of classloaders to be used with a particular software application (see for example, p.4, paragraph [0051], "where each manifest comprises an XML file").

Claim 13:

Taylor also discloses the method of claim 12, wherein said control file is a deployment descriptor (see for example, p.4, paragraph [0048], "The population manifest 326 includes information on components to load in the container 306" and paragraph [0051], "where each manifest comprises an XML file").

Claim 14:

Taylor further discloses the method of claim 13, wherein said control file is interpreted by an application container constructor during deployment so as to define the application container (see for example, p.4, paragraph [0050], "The population manifest 326 is parsed by the container 306 to construct class loader...").

Claim 16:

Taylor also discloses the method of claim 11, wherein said hierarchy is specified by a classloader structure declaration (see for example, Fig.6, an example of a class loader hierarchy and related text description in the specification).

Claim 17:

Taylor further discloses the method of claim 11, wherein a combination of said modules may be associated with a plurality of subordinate classloaders (see for example, p.4, paragraph [0051], "root chain", "The root chain is built first, then the class loaders used to load the factories and then the class loaders used to load the components").

Claim 19:

Taylor further discloses the method of claim 11, wherein the server provides multiple software applications, each with their own hierarchy of classloaders (see for example, p.4, paragraph [0049], hierarchy of class loads to be employed in loading class files associated with components listed in the manifest").

Claim 33:

Taylor also discloses the method of claim 11 wherein each of the plurality of modules is one of an EJB or Web application file, together with associated classes, configuration rules and resource files fro that EJB or Web application file (see for example, Fig.2, structure of 50 of a CAR file and related text; also see

paragraph [0030], The CAR file structure 50 includes the following elements, implementation resources, ad download JAR file, a Security policy and a JAR manifest).

Claim 34:

Taylor further discloses the method of claim 11 wherein the hierarchy that includes a plurality of branches specified by the software developer further comprises a plurality of nested references to modules and/or individual class files as specified by the software developer application (see for example, p.4, paragraph [0048], "An administrator would place... and provide the population manifest 326 indicating the components that will be loaded"; also see Fig.6, Fig.7 and related text).

Claim 44:

Taylor discloses the method of Claim 11, wherein the control file includes a tag layout, and application class-loader structure elements, that determine the hierarchy of modules and classes of the software application to be loaded into the application server (see for example, Fig.7, XML file example - tag layout and elements and hierarchy of class, components, factory..."); and wherein upon deployment, the deployment utility retrieves modules and classes from a computer readable medium in a manner consistent with the tag layout in the configuration file, and constructs an application container at the server with the

classes and modules, in the order in which the classes and modules were retrieved, to create a hierarchical classloader (see for example, Fig.8, step 402, starts class sever, step 404, load root classes; step 406, load factories...)

Claims 1-4, 6, 7, 9, 31-32 and 43

Claims 1-4, 6, 7, 9, 31-32 and 43are system version for performing the claimed method as in claims 11-14, 16, 17, 19, 33, 34 and 44 addressed above, wherein all claimed limitation functions have been addressed and/or set forth above and certainly a computer system would need to run and/or practice such function steps disclosed by reference above. Thus, they also would have been obvious.

Claims 45 and 46:

Claims 45 and 46 are computer program products version of the claimed method, wherein all claimed limitation functions have been addressed in claims 11 and 44 above respectively. It is well known in the computer art that such method steps can be implemented as computer program and can be practiced and /or stored on a computer operable media. Thus, they also would have been obvious in view of reference teachings above.

Conclusion

9. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

10. Applicant's arguments with respect to claims rejection have been considered but are moot in view of the new grounds of rejection and Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a). A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Zheng Wei whose telephone number is (571) 270-1059 and Fax number is (571) 270-2059. The examiner can normally be reached on Monday-Thursday 8:00-15:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature of relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571- 272-1000.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Z. W./
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192